William Stallings Computer Organization and Architecture

Chapter 10
RISC vs CISC

Introdução

- **RISC** = **R**educed **I**nstruction **S**et **C**omputer
- Elementos básicos:
  - Grande número de registradores de propósito geral ou uso de tecnologias de compilação na otimização do uso de registradores
  - Conjunto limitado (reduzido) de instruções simples
  - Enfoque na utilização de pipeline de instruções
- Arquitetura oposta à **CISC** (Complex **I**nstruction **S**et **C**omputer)

Introdução

- A CISC (em inglês: *Complex Instruction Set Computing*, Computador com um Conjunto Complexo de Instruções), usada em processadores Intel e AMD; suporta mais instruções no entanto, com isso, mais lenta fica a execução delas.
- A RISC (em inglês: *Reduced Instruction Set Computing*, Computador com um Conjunto Reduzido de Instruções) usada em processadores PowerPC (da Apple, Motorola e IBM) e SPARC (SUN); suporta menos instruções, e com isso executa com mais rapidez o conjunto de instruções que são combinadas.

Introdução

Um processador CISC (Complex Instruction Set Computer), é capaz de executar várias centenas de instruções complexas, sendo extremamente versátil. Exemplos de processadores CISC são os 386 e os 486. No começo da década de 80, a tendência era construir chips com conjuntos de instruções cada vez mais complexos, mas alguns fabricantes resolveram seguir o caminho oposto, criando o padrão RISC (Reduced Instruction Set Computer).

Introdução

Ao contrário dos complexos CISC, os processadores RISC são capazes de executar apenas algumas poucas instruções simples. Justamente por isso, os chips baseados nesta arquitetura são mais simples e muito mais baratos. Outra vantagem dos processadores RISC, é que por terem um menor número de circuitos internos, podem trabalhar com clocks mais altos. Um processador RISC é capaz de executar instruções muito mais rapidamente.

CISC

CISC (Complex Instruction Set Computer, ou, em uma tradução literal, “Computador com um Conjunto Complexo de Instruções”): é um processador capaz de executar centenas de instruções complexas diferentes sendo, assim, extremamente versátil. Exemplos de processadores CISC são os 386 e os 486 da Intel.

CISC

Os processadores baseados na computação de conjunto de instruções complexas contêm uma micro programação, ou seja, um conjunto de códigos de instruções que são gravados no processador, permitindo-lhe receber as instruções dos programas e executá-las, utilizando as instruções contidas na sua micro programação. Seria como quebrar estas instruções, já em baixo nível, em diversas instruções mais próximas do hardware (as instruções contidas no microcódigo do processador). Como característica marcante esta arquitetura contém um conjunto grande de instruções, a maioria deles em um elevado grau de complexidade.

CISC

Examinando do ponto de vista um pouco mais prático, a vantagem da arquitetura CISC é que já temos muitas das instruções guardadas no próprio processador, o que facilita o trabalho dos programadores de linguagem de máquina; disponibilizando, assim, praticamente todas as instruções que serão usadas em seus programas. Os processadores CISC têm a vantagem de reduzir o tamanho do código executável por já possuírem muito do código comum em vários programas, em forma de uma única instrução.

CISC

A CISC é implementada e guardada em micro-código no processador, sendo difícil modificar a lógica de tratamento de instruções. Esta arquitetura suporta operações do tipo “a=a+b” descrita por “add a,b”, ou seja podem simplesmente utilizar **dois operandos** para uma única instrução, sendo um deles fonte e destino (acumulador) e permite um ou mais operadores em memória para a realização das instruções. Com isto se comprova a necessidade de abranger um elevado leque de modelos de endereçamento, com acesso direto à memória e com apontadores para as variáveis em memória, armazenados eles próprios (ponteiros) em células de memória.

CISC

Do ponto de vista da performance, os CISC's têm algumas desvantagens em relação aos **RISC's**, entre elas a impossibilidade de se alterar alguma instrução composta para se melhorar a performance. O código equivalente às instruções compostas do CISC pode ser escrito nos RISC's da forma desejada, usando um conjunto de instruções simples, da maneira que mais se adequar. Sendo assim, existe uma disputa entre tamanho do código X desempenho.

RISC

Reduced Instruction Set Computer ou Computador com um Conjunto Reduzido de Instruções (RISC), é uma linha de arquitetura de computadores que favorece um conjunto simples e pequeno de instruções que levam aproximadamente a mesma quantidade de tempo para serem executadas. Como exemplo de aplicação dessa arquitetura são DEC Alpha, SPARC, MIPS, e PowerPC. O tipo de microprocessador mais largamente usado em desktops, o x86, é mais CISC do que RISC, embora chips mais novos traduzam instruções x86 baseadas em arquitetura CISC em formas baseadas em arquitetura RISC mais simples, utilizando prioridade de execução.

RISC

Os processadores baseados na computação de conjunto de instruções reduzido não tem micro-programação, as instruções são executadas diretamente pelo hardware. Como característica, esta arquitetura, além de não ter microcódigo, tem o conjunto de instruções reduzido, bem como baixo nível de complexidade.

RISC

Para garantir rapidez e eficiência do sistema, pretende-se que os operadores sejam acendidos à velocidade de funcionamento do processador, logo se justifica a utilização dos registradores, e, para que a representação de todas as variáveis para processamento sejam apresentadas como registros, tem que se garantir um número elevado destes, assegurando-se atualmente a maioria das variáveis escalares pela utilização de 32 registros genéricos que caracterizam a maioria da tecnologia dos compiladores atuais.

VLSI

Em termos de VLSI (*Very Large Scale Integration*) a tecnologia da altura apenas permitia densidades de transístores que seriam muito baixas quando comparadas com os *standards* de hoje. Era simplesmente impossível colocar muitas funcionalidades num único *chip*. No início dos anos 80, quando se começou a desenvolver a arquitectura RISC, um milhão de transístores num único *chip* era já bastante . Devido à falta de recursos (transístores) as máquinas CISC da altura tinham as suas unidades funcionais espalhadas por vários *chips*. Isto era um problema por causa do alto tempo de espera nas transferências de dados entre os mesmos, o que desde logo era um óbice ao desempenho. Uma implementação num único *chip* seria o ideal.

VLSI

CARACTERÍSTICA	Processador	
	80486 (Intel, CISC)	SPARC (Sun, RISC)
Ano de desenvolvimento	1989	1987
Número de instruções	235	69
Tamanho de uma instrução (Bytes)	1-11	4
Modos de endereçamento	11	1
Número de registradores de propósito geral	8	40-520

Característica da Execução de Instruções

- Com as linguagens de programação de baixo nível, o custo da produção de SW era bem alto
- Além disso, falhas de SW eram inerentemente mais comuns que falhas de HW
- Esse panorama mudou com o surgimento das linguagens de programação de alto nível
  - Elas permitem ao programador abstrair os detalhes da máquina e expressar algoritmos de forma mais concisa

Característica da Execução de Instruções

- Outros pesquisadores tomaram o caminho inverso: simplificar o conjunto de instruções
- Eles tomaram com base estudos que determinavam as características e os padrões de execução de instruções de máquina geradas por programas de alto nível
- Os aspectos examinados foram:
  - Operações realizadas
  - Operandos usados
  - Chamadas de procedimento

Característica da Execução de Instruções - Operações

	Ocorrência dinâmica		Ponderada por instrução de máquina		Ponderada por referência à memória	
	Pascal	C	Pascal	C	Pascal	C
Comando de atribuição	45	38	13	13	14	15
Comando de repetição	5	3	42	32	33	26
Chamada de proced.	15	12	31	33	44	45
Comando condicional	29	43	11	21	7	13
Desvio incondicional	-	3	-	-	-	-
Outros	6	1	3	1	2	1

São executados mais freqüentemente

Consomem mais tempo na execução

Característica da Execução de Instruções - Operandos

	Pascal	C	Média
Constante inteira	16	23	20
Variável escalar	58	53	55
Vetor/registro	26	24	25

- A maioria das referências é para variáveis escalares - e locais
  - Referências a vetores ou registros incluem referências anteriores a seus apontadores, que são, também, variáveis escalares locais

Característica da Execução de Instruções - Procedimentos

- Aspectos significativos:
  - Número de parâmetros e variáveis utilizados

% de execução de chamadas de proc. com:	Compiladores, interpretadores e editores de texto	Pequenos programas não numéricos
> 3 argumentos	0-7 %	0-5 %
> 5 argumentos	0-3 %	0 %
> 8 argumentos e variáveis escalares locais	1-20 %	0-6 %
> 12 argumentos e variáveis escalares locais	1-6 %	0-3 %

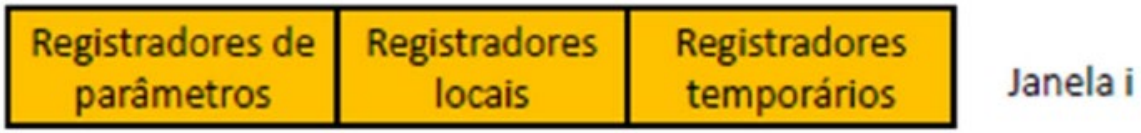
  - Nível de aninhamento de procedimentos
    - Normalmente, o programa permanece limitado a uma janela de profundidade limitada

Característica da Execução de Instruções - Resumo

- Comandos de atribuição (movimentação de dados) são muito utilizados
- Esses dados são, em sua maioria, variáveis escalares locais
- Abordagens visando melhoria do desempenho
  - Abordagem baseada em HW
    - Banco de Registradores
  - Abordagem baseada em SW
    - Técnicas de compilação que visam otimizar a utilização de registradores

Banco de Registradores

- Registradores são organizados em **janelas** para diminuir o acesso à memória



- Cada janela é dedicada a uma chamada de procedimento
- Tipos de registradores em uma janela
  - Registradores de parâmetros
  - Registradores locais
  - Registradores temporários

Banco de Registradores

- Os registradores temporários armazenam os parâmetros passados para o procedimento
- Os registradores locais contêm as variáveis locais ao procedimento
- Os registradores temporários são usados para trocar resultados/parâmetros com o procedimento seguinte

Variáveis Globais

- Usadas por mais de um procedimento
- Duas alternativas para armazenamento
  - As variáveis globais são armazenadas na memória
    - Abordagem simples ☺
    - Ineficiente para variáveis usadas com freqüência ☹
  - Variáveis globais são armazenadas em registradores específicos no µP
    - Abordagem mais eficiente ☺
    - Mais complexa que a anterior ☹
      - HW mais complexo; mecanismos de endereçamento diferentes
      - Compilador deve decidir quais variáveis devem ser armazenadas nos registradores

Banco de Registradores versus Memória Cache

Banco de Registradores	Memória Cache
Todas as variáveis escalares locais	Variáveis escalares locais usadas recentemente
Variáveis individuais	Blocos de memória
Variáveis globais designadas pelo compilador	Variáveis globais usadas recentemente
Operações de salvamento/restauração baseadas na profundidade de aninhamento de procedimentos	Operações de salvamento/restauração baseadas no algoritmo de substituição da cache
Endereçamento de registrador	Endereçamento de memória

Otimização do Uso de Registradores por Compiladores

- Suposições:
  - Há apenas um pequeno conjunto de registradores
  - O objetivo do compilador é armazenar os operandos usados mais freqüentemente nesses registradores, minimizando o acesso à memória
- Cada operando candidato a residir em um desses registradores é associado a um registrador virtual
- O problema é definir quais destes (potencialmente) vários registradores virtuais devem ser associados aos (relativamente) poucos registradores reais

Características de Arquiteturas RISC

- Uma instrução por ciclo de máquina
  - Um ciclo de máquina é o tempo requerido para buscar dois operandos em registradores, executar uma operação da ULA e armazenar o resultado em um registrador
- Operações de registrador para registrador
  - Com operações simples de CARGA e ARMAZENAMENTO
- Modos de endereçamento simples
- Formatos de instrução simples

Controvérsia RISC x CISC

- Critérios de avaliação da arquitetura RISC
  - Quantitativos
    - Comparação de tamanho e velocidade entre programas executáveis gerados em máquinas RISC e CISC
  - Qualitativos
    - Exame de parâmetros tais como suporte a linguagens de alto nível e uso ótimo da VLSI
- Entusiastas da arquitetura CISC refutaram os critérios apresentados acima

Controvérsia RISC x CISC

- Alguns contra-argumentos apresentados
  - Não há um conjunto definitivos de programas a serem utilizados para teste
  - É difícil distinguir os efeitos gerados pelo HW daqueles gerados pela maneira como o compilador foi escrito
- Hoje, tem-se a **convergência** entre RISC e CISC
  - Densidade de chip crescente faz com que os µP fiquem mais complexos (característica CISC)
  - Demanda por melhor desempenho faz com que haja um aumento do número de registradores de uso geral e na otimização do pipeline de instruções (característica RISC)

Modelos Híbridos

Apesar de por questões de Marketing, muitos fabricantes venderem seus chips, como sendo "Processadores RISC", não existe praticamente nenhum processador atualmente que siga estritamente uma das duas filosofias. Tanto processadores da família x86, como o Pentium II, Pentium III e AMD Athlon, quanto processadores supostamente RISC, como o MIPS R10000 e o HP PA-8000, ou mesmo o G4, utilizado nos Macintosh, misturam características das duas arquiteturas, por simples questão de performance.

Modelos Híbridos

Examinando de um ponto de vista um pouco mais prático, a vantagem de uma arquitetura CISC é que já temos muitas das instruções guardadas no próprio processador, o que facilita o trabalho dos programadores, que já dispõe de praticamente todas as instruções que serão usadas em seus programas. No caso de um chip estritamente RISC, o programador já teria um pouco mais de trabalho, pois como disporia apenas de instruções simples, teria sempre que combinar várias instruções sempre que precisasse executar alguma tarefa mais complexa. Seria mais ou menos como se você tivesse duas pessoas, uma utilizando uma calculadora comum, e outra utilizando uma calculadora cientifica. Enquanto estivessem sendo resolvidos apenas cálculos simples, de soma, subtração, etc. quem estivesse com a calculadora simples poderia até se sair melhor, mas ao executar cálculos mais complicados, a pessoa com a calculadora científica disporia de mais recursos

Modelos Híbridos

A ideia de construção de um processador híbrido é bastante interessante, pois faz com que finalmente PCs possam ter um desempenho realmente astronômica. A Intel, porém, errou feio em um detalhe importante do projeto do Pentium Pro: o seu decodificador CISC foi desenvolvido basicamente para trabalhar com código de 32 bits – ou seja, com sistemas operacionais como o Windows NT, OS/2 e Netware. Todos nós sabemos que a maioria dos usuários ainda trabalha com sistemas operacionais de 16 bits como o MS-DOS, Windows 3.x e Windows 95.

Isto quer dizer que, se tivermos um Pentium-200 e um Pentium Pro-200, um Windows 3.11 será mais rápido no Pentium e não no Pentium Pro, por mais incrível que possa parecer.

Modelos Híbridos

Não valeria a pena adquirir um micro baseado no Pentium Pro se você fosse utilizar MS-DOS, Windows 3.x ou Windows 95. Processadores de outros fabricantes – em especial o 6×86 da Cyrix e o 5K86 da AMD – também possuem arquitetura híbrida CISC/RISC, com a vantagem de possuírem um decodificador otimizado para código tanto de 32 bits quanto de 16 bits. Nos chips atuais, que são na verdade misturas das duas arquiteturas, juntamos as duas coisas. Internamente, o processador processa apenas instruções simples. Estas instruções internas, variam de processador para processador, são como uma luva, que se adapta ao projeto do chip.

Modelos Híbridos

As instruções internas de um K6 são diferentes das de um Pentium por exemplo. Sobre estas instruções internas, temos um circuito decodificador, que converte as instruções complexas utilizadas pelos programas em várias instruções simples que podem ser entendidas pelo processador. Estas instruções complexas sim, são iguais em todos os processadores usados em micros PC. É isso que permite que um Athlon e um Pentium III sejam compatíveis entre sí.

Modelos Híbridos

O conjunto básico de instruções usadas em micros PC é chamado de conjunto x86. Este conjunto é composto por um total de 187 instruções, que são as utilizadas por todos os programas. Além deste conjunto principal, alguns processadores trazem também instruções alternativas, que permitem aos programas executar algumas tarefas mais rapidamente do que seria possível usando as instruções x86 padrão. Alguns exemplos de conjuntos alternativos de instruções são o MMX (usado a partir do Pentium MMX), o 3D-NOW! (usado pelos processadores da AMD, a partir do K6-2), e o SSE (suportado pelo Pentium III).